The event viewer library

This document describes the event viewer library which can be used to view timing events of the emBODY using Vision.

Approval History

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Version | Author | | Date | Approved | | Date |
| 1.0 | Accame | iCub Unit |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 0.1 | 29 Feb 12 | Accame | First version. |
| 1.0 | 08 Jan 13 | Accame | Changed according to new folder structure. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1 Introduction 1](#_Toc376941925)

[2 How to configure it 2](#_Toc376941926)

[2.1 Asynchronous mode 3](#_Toc376941927)

[2.2 Synchronous mode 5](#_Toc376941928)

[3 How to use it 6](#_Toc376941929)

[4 What to see 7](#_Toc376941930)

[4.1 Use with OOSIIT 7](#_Toc376941931)

[4.2 Use with direct calls to eventviewer APIs 7](#_Toc376941932)

# Introduction

The event viewer can be used to view timing events from every software module of emBODY.

The timing events can come from the OOSIIT (and hence OSAL) when compiled in debug mode or from explicit calls of the user.

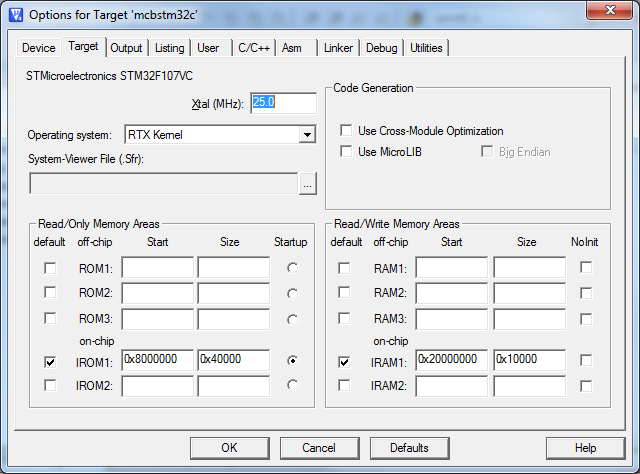
Events originated by OOSIIT are those related to task switching of the scheduler and from the activity of the three ISR: SysTick, SVC, pendSV.

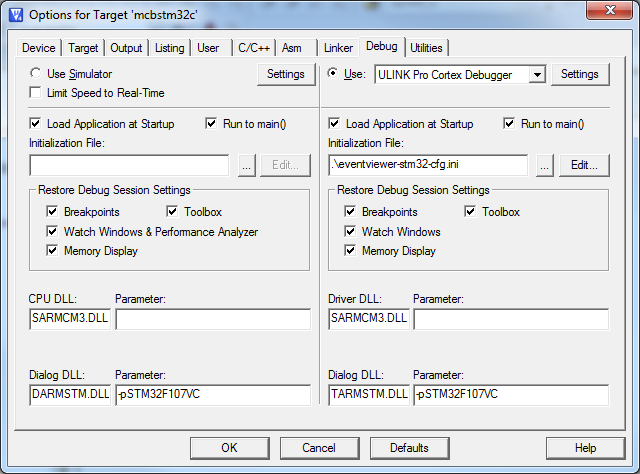
Events can also be originated by the user with explicit calls to eventviewer APIs for the sake of displaying timing in runtime.

How to configure it

In Vision follow the steps:

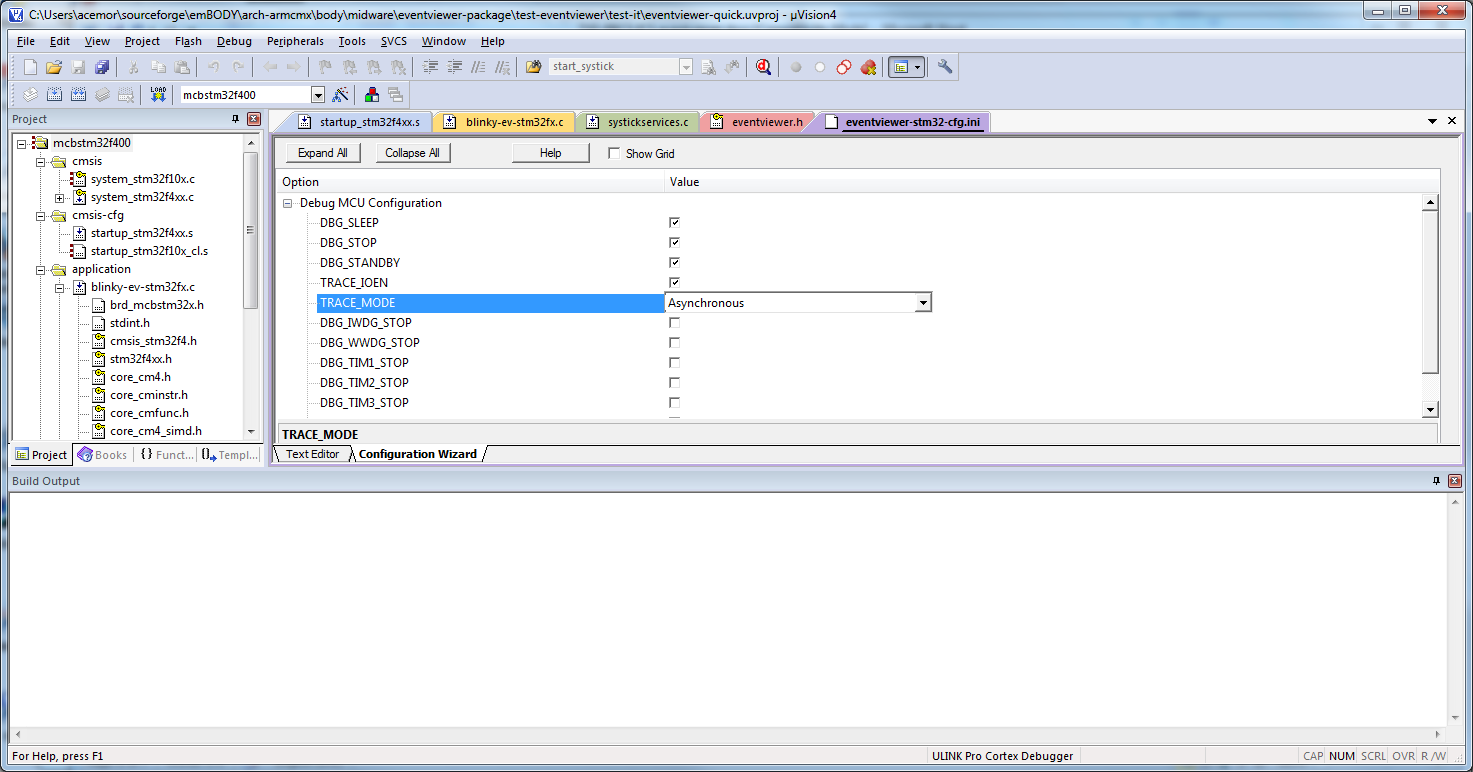
1. Include the eventviewer.c in compilation (or a library) and set the include path of eventviewer.h,
2. In options-target specify the use of RTX Kernel,
3. In options-debug use the ULINK-PRO, use the file *eventviewer-stm32-cfg.ini* (or similar w/ correct TRACE\* initialisation),
4. In options-debug-ULINKPRO-settings, specify as in figures.

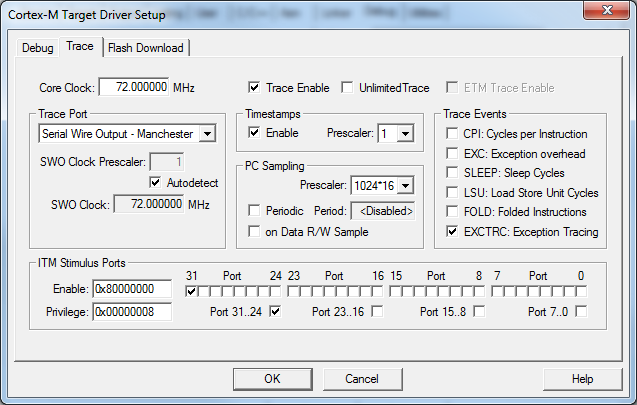




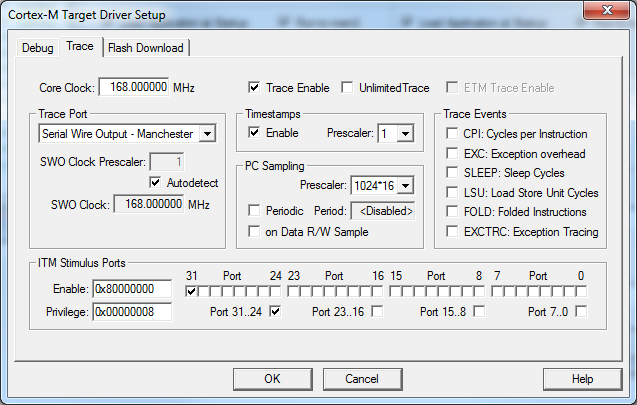
## Asynchronous mode

It works for both STM32F107 and STM32F407

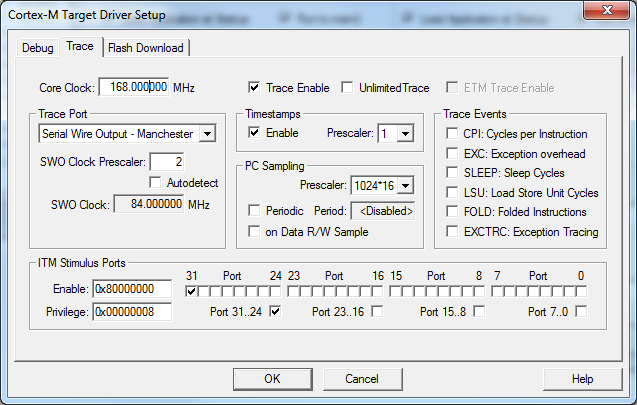




Use 72 MHz for STM32F107.



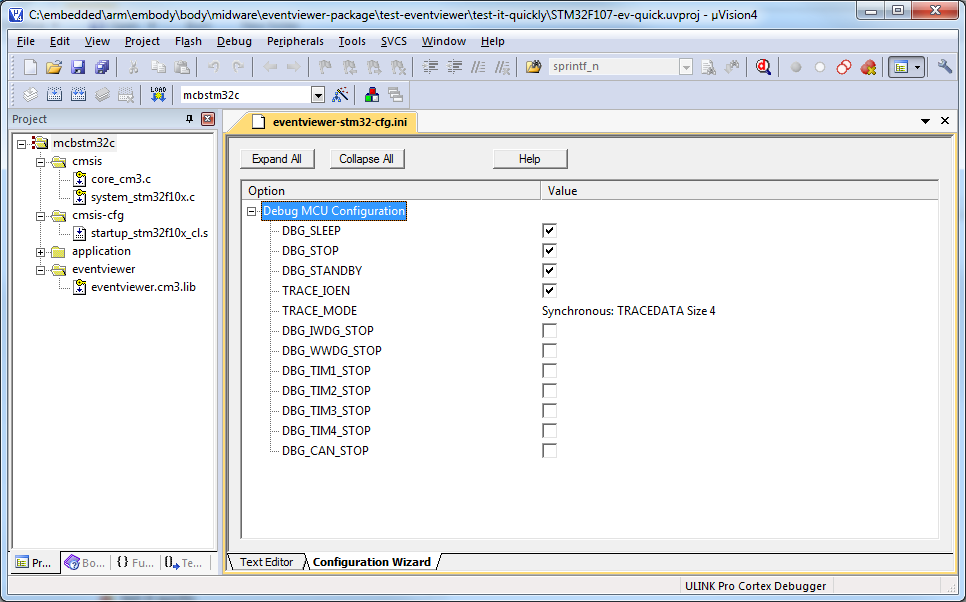
Use 168 MHz for STM32F107 in autodetect mode.

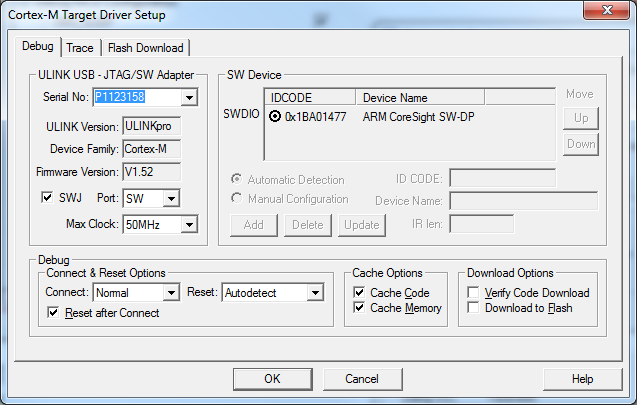


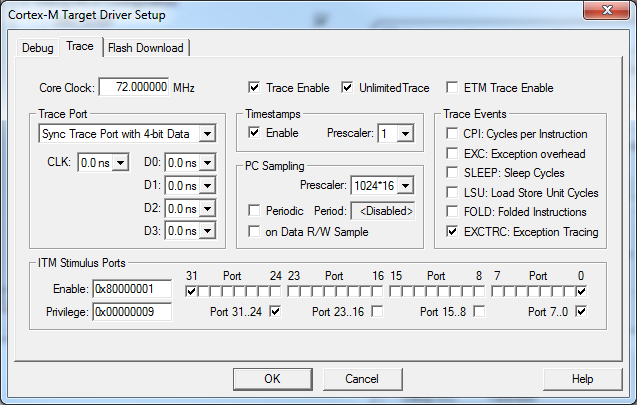
Or even use a prescaler.

## Synchronous mode

It is OK for STM32F107, but not always. It does not work for STM32F407.

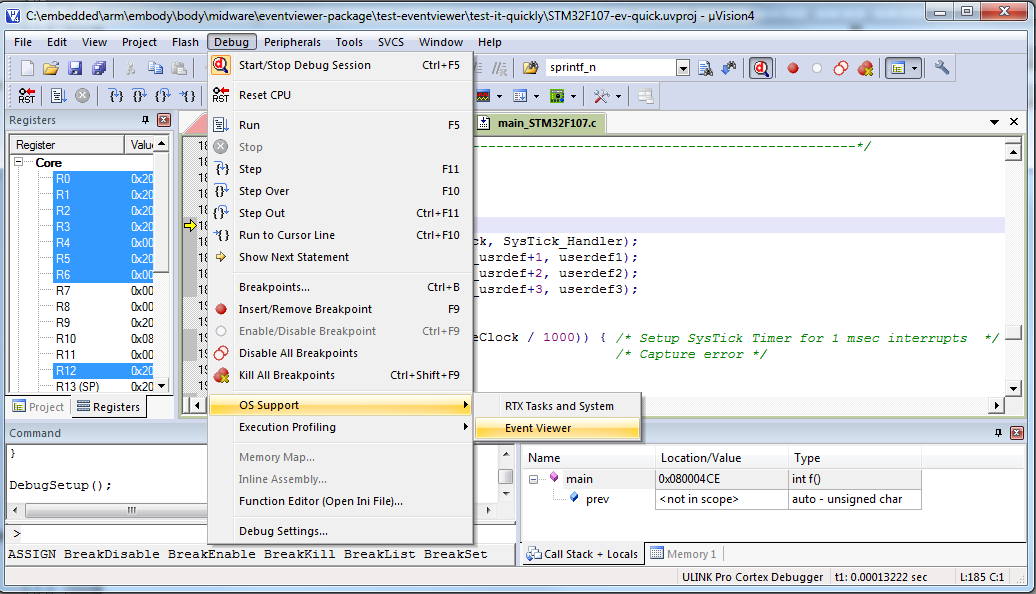






How to use it

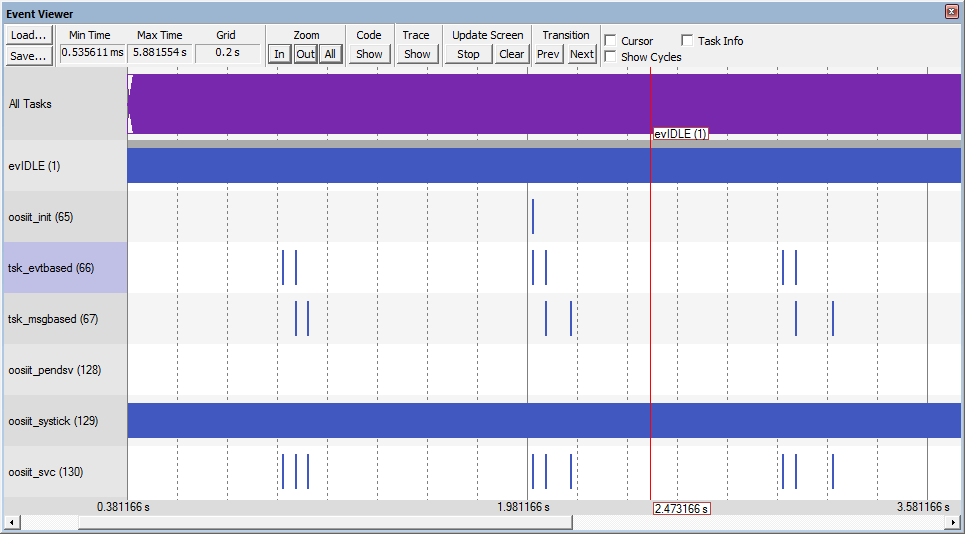
Compile, attach the ULINK-pro, run debugger and enable Event Viewer.



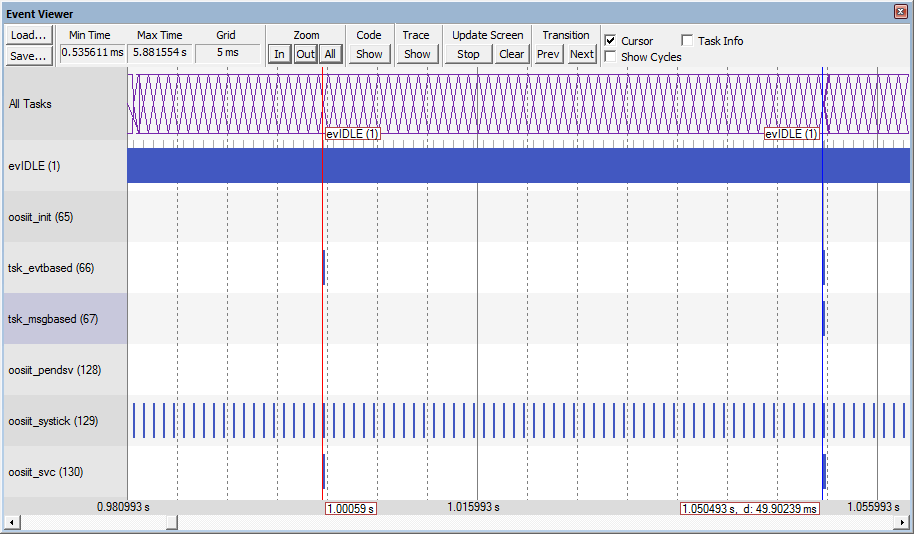
What to see

## Use with OOSIIT

In case it is used the OOSIIT compiled in debug mode the diagram will be related to task switching and OOSIIT internal events. The following is a snapshot of the test project of OOSIIT.



**Figure 1**: Snapshot of project /oosiit-tests/test-it/oosiit-full.uvproj. All the tasks are in not active most of the time. They periodically wake up an trigger some actions.



**Figure 2**: Snapshot of project /oosiit-tests/test-it/oosiit-full.uvproj. Task evtbased receives an event every second from the systick. It switches the LED on, then it waits for 50 msec. It switches the LED off and then sends a message to task msgbased which is soon activated.

## Use with direct calls to eventviewer APIs

The code is in following figure.

The code with direct calls to eventviewer

void idle(void);

void userdef1(void);

void userdef2(void);

void userdef3(void);

int main(void)

{

evEntityId\_t prev;

eventviewer\_init();

eventviewer\_load(ev\_ID\_idle, idle);

eventviewer\_load(ev\_ID\_systick, SysTick\_Handler);

eventviewer\_load(ev\_ID\_first\_usrdef+1, userdef1);

eventviewer\_load(ev\_ID\_first\_usrdef+2, userdef2);

// the eventviewer shall stay most of time in idle

// apart from some specific actions: systick, userdef1 and userdef2

eventviewer\_switch\_to(ev\_ID\_idle);

board\_led\_init();

systickserv\_start\_systick(1000, myonsystick);

for(;;)

{

prev = eventviewer\_switch\_to(ev\_ID\_first\_usrdef+1);

board\_led\_on(board\_led\_0);

eventviewer\_switch\_to(prev);

systickserv\_wait\_for(500\*1000);

prev = eventviewer\_switch\_to(ev\_ID\_first\_usrdef+2);

board\_led\_off(board\_led\_0);

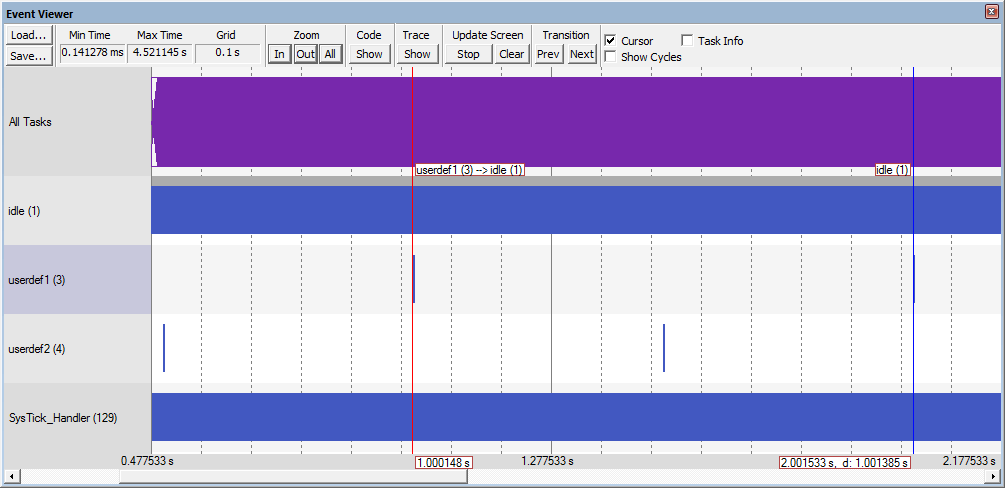
eventviewer\_switch\_to(prev);

systickserv\_wait\_for(500\*1000);

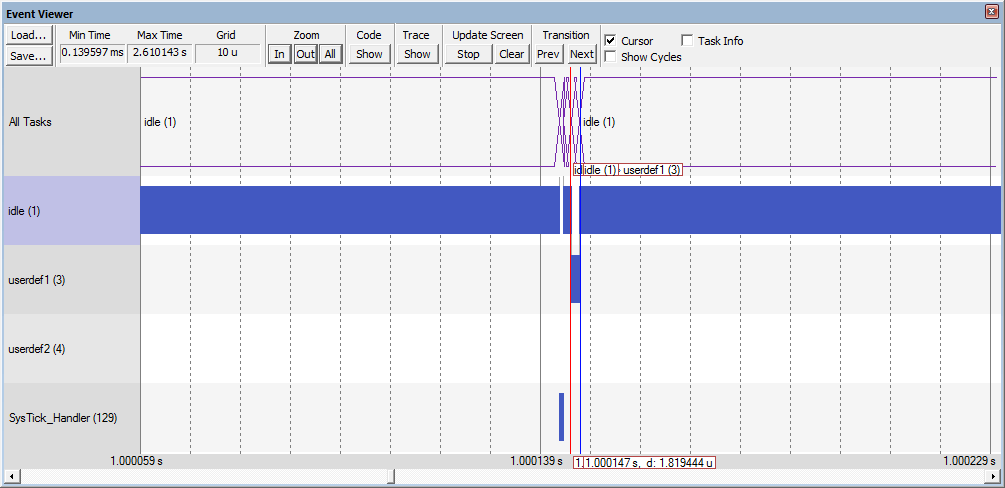
}

}

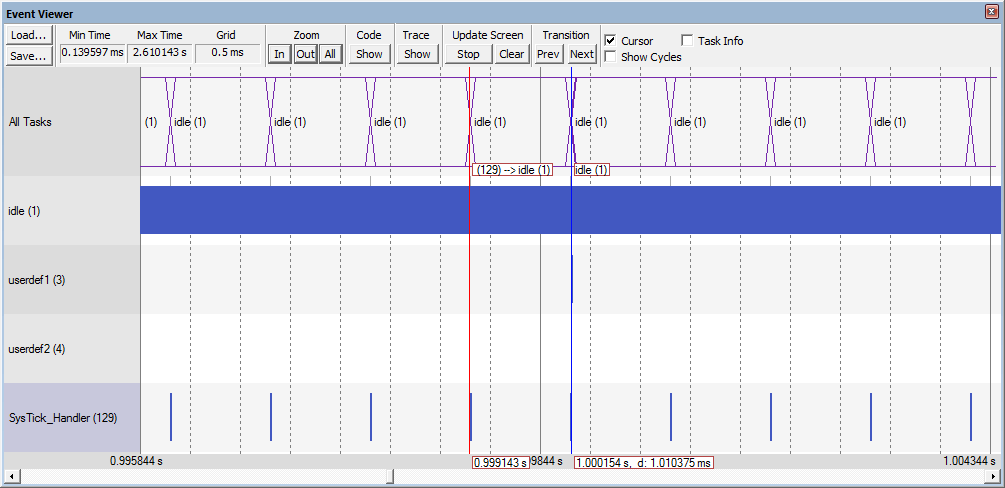
**Figure 3**: The code with direct calls to eventviewer.



**Figure 4**: Snapshot of project /eventviewer-tests/test-it/eventviewer-full.uvproj. The userdef1 signal triggers every 1 sec and lasts for 1.8 sec.



**Figure 5**: Snapshot of project /eventviewer-tests/test-it/eventviewer-full.uvproj. The userdef1 signal lasts for 1.8 sec.



**Figure 6**: Snapshot of project /eventviewer-tests/test-it/eventviewer-full.uvproj. The systick triggers every 1 msec.